Formalising Bipermutative Categories and Quantum Program Semantics in Agda

Malin Altenmüller, Robin Kaarsgaard

Context

String diagrams model *strict* monoidal categories: Structural equations do not hold computational content. In a proof assistant like Agda, modelling this behaviour implicitly is highly non-trivial.

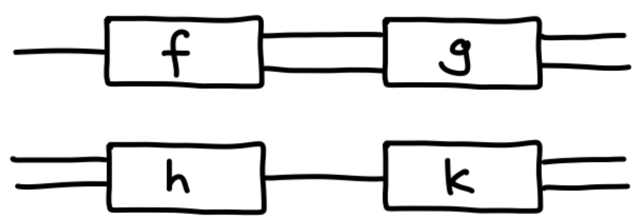
Quantum Computing via Rig Categories

Quantum programs can be modelled by rig categories (together with certain effects) which contain a large number of coherence equations.

Example 1: Interchange Law

$$(f \mathbin{⨾} g) \otimes (h \mathbin{⨾} k) = (f \otimes h) \mathbin{⨾} (g \otimes k)$$

Both sides of this equation are represented by the same string diagram:



How to model this in the formalisation?

Goal

Agda implementation modelling string diagrams in which all coherence isomorphisms are implicit.

Agda's Rewrite Rules

- Adding user-specified definitional equalities to Agda.
- Rules are automatically applied wherever possible.
- Plan: add all coherence equations as rewrite rules.

Coherence Isomorphisms as Rewrite Rules

```
assocr : {A B C : Ob} → (A ⊗ B) ⊗ C ↔ A ⊗ B ⊗ C
unitl : (A : Ob) → (one ⊗ A) ↔ A
unitr : (A : Ob) → (A ⊗ one) ↔ A

{-# REWRITE assocr unitl unitr #-}
```

Example 2: Implementation of a Property of the Language $\sqrt{\Pi}$

```
lem : ctrl z ∘ s ⊗C id two ⇔ (s ⊗C id two) ∘ ctrl z
lem = ctrl (p -one) ∘ s ⊗C id two                          – (1)
  =[ cong (λ x → x ∘ s ⊗C id two) (sym (A-1 -one)) ]>
    swap ∘ ctrl (p -one) ∘ swap ∘ (s ⊗C id two) – (2)
  =[ cong (λ x → swap ∘ ctrl (p -one) ∘ x) (swap⊗C {c1 = s}{id two}) ]>
    swap ∘ ctrl (p -one) ∘ (id two ⊗C s) ∘ swap – (3)
  =[ cong (λ x → swap ∘ x ∘ swap) (sym (A-2 -one i)) ]>
    swap ∘ (id two ⊗C s) ∘ ctrl z ∘ swap           – (4)
  =[ cong (λ x → x ∘ ctrl z ∘ swap) (swap⊗C {c1 = id two}{s}) ]>
    (s ⊗C id two) ∘ swap ∘ ctrl z ∘ swap           – (5)
  =[ cong (λ x → (s ⊗C id two) ∘ x) (A-1 -one) ]>
    (s ⊗C id two) ∘ ctrl z []                      – (6)
```

Proof by Hand

The formalisation of the proof of an example property of $\sqrt{\Pi}$ (see left) has exactly as many steps as the proof on paper:

$$
\begin{aligned}
&\mathsf{Ctrl}\ Z \circ (\mathsf{S} \otimes \mathsf{Id}) && (1)\\
&= \mathsf{SWAP} \circ \mathsf{Ctrl}\ Z \circ \mathsf{SWAP} \circ (\mathsf{S} \otimes \mathsf{Id}) && (2)\\
&= \mathsf{SWAP} \circ \mathsf{Ctrl}\ Z \circ (\mathsf{Id} \otimes \mathsf{S}) \circ \mathsf{SWAP} && (3)\\
&= \mathsf{SWAP} \circ (\mathsf{Id} \otimes \mathsf{S}) \circ \mathsf{Ctrl}\ Z \circ \mathsf{SWAP} && (4)\\
&= (\mathsf{S} \otimes \mathsf{Id}) \circ \mathsf{SWAP} \circ \mathsf{Ctrl}\ Z \circ \mathsf{SWAP} && (5)\\
&= (\mathsf{S} \otimes \mathsf{Id}) \circ \mathsf{Ctrl}\ Z && (6)
\end{aligned}
$$

Example 3: Triangle Equation

$$
\begin{tikzcd}
(A \otimes 1) \otimes B \arrow[rr, "\alpha_{A,1,B}"] \arrow[dr, "\rho_A \otimes 1_B"'] & & A \otimes (1 \otimes B) \arrow[dl, "1_A \otimes \lambda_B"] \\
& A \otimes B &
\end{tikzcd}
$$

Triangle Equation in Agda

```
triangle-eq : {A B : Ob} → unit⊗r A ⊗ id B
                ⇔ assoc⊗r {A}{one}{B} ⨾ (id A ⊗ unit⊗l B)
triangle-eq = id
```

Alternative Solution

Instead, we may specify coherence isomorphisms as *propositional* equalities: This introduces explicit equations, but we can use Agda's metatheory to apply them automatically.

```
assocr' : {A B C : Ob} → (A ⊗ B) ⊗ C ≡ A ⊗ (B ⊗ C)
```

Contact

malin.altenmuller@ed.ac.uk